Sentiment Analysis using Naïve Bayes' with Hadoop

Shabbir Saifee and Shubham Deb

College of Computer and Information Science

Northeastern University

s.saifee@husky.neu.edu, deb.sh@husky.neu.edu

1. Abstract

The advent of Internet has led to copious amount of sentimental content available on the Web. Such content is often found in social media web sites in the form of movie or product reviews, user comments, testimonials, messages in discussion forums etc. Sentiment analysis or opinion mining is the identification of subjective information from text. This project involves classification of reviews/comments into two main sentiments: positive and negative. The paper elaborately discusses how to implement sentiment analysis on text using the naive Bayes classification algorithm available on Apache Mahout.

2. Introduction

This section will cover the overall aim of the project and the motivation behind it.

Objective

The overall goal of this project was to perform a sentiment analysis on opinions/reviews/comments given in various social networking sites e.g. twitter. The Sentiment found within comments, feedback or critiques provide useful indicators for many different purposes and can be categorized by polarity. By polarity we tend to find out if a review is overall a positive one or a negative one. For example:

1. Positive Sentiment in subjective sentence: "I loved the movie Inception"—This sentence is expressed positive sentiment about the movie Inception and we can decide that from the sentiment threshold value of word "loved". So, threshold value of word "loved" has positive numerical value.
2. Negative sentiment in subjective sentences: "Cat woman is a flop movie" defined sentence is expressed negative sentiment about the movie and we can decide that from the sentiment threshold value of word "flop". So, threshold value of word "flop" has negative numerical value.

Motivation

Over the past decade humans have experienced exponential growth in the use of online resources, social media and microblogging websites such as Twitter, Facebook and YouTube. Many companies and organizations have identified these resources as a rich mine of marketing knowledge. Traditionally companies used interviews, questionnaires and surveys to gain feedback and insight into how customers felt about their products. These traditional methods were often extremely time consuming and expensive and did not always return the results that the companies were looking for due to environmental factors and poorly designed surveys.

Natural language processing and sentiment analysis are playing an increasingly important role in making educated decisions on Marketing Strategies and giving valuable feedback on products and services. There are massive amounts of data containing consumer sentiment uploaded to the internet every day, this type of data is predominantly unstructured text that is difficult for computers to gain meaning from. In the past it was not possible to process such large amounts of unstructured data but now with computational power following the projections of Moore's law (Moore, 1965) and distributed networks of computers using frameworks such as Hadoop, massive datasets can be now processed with relative ease. Major investment is going into this area such as IBMs tireless research into their Natural Language Processing supercomputer Watson and Googles recent acquisition of deep mind technology. With further research and investment into this area machines will soon be able to gain an "understanding" from text which will

greatly improve data analytics and search engines. The use of sentiment analysis in a commercial environment is growing. This is evident in the increasing number of brand tracking and marketing companies offering this service. Some services include:

- To prevent viral effects.
- Assessing market buzz, competitor activity and customer trends, fads and fashion.
- Measuring public response to an activity or company related issue.

3. Background

This section aims to give an overview of the background material used for this project.

Sentiment analysis and Data Mining

Data mining is the computational process of finding patterns in large datasets and its methods are at the intersection between Artificial Intelligence, Machine Learning, computer science, data base technologies, and statistics. The objective of data mining is to extract information or knowledge from a dataset and transform it into a structure that can be understood.

Sentiment analysis is the field of study that analyses people's opinions, sentiments, evaluations, appraisals, attitudes, and emotions towards entities such as products, services, organizations, and their attributes.

Sentiment mining is one of the important aspects of data mining where important data can be mined based on the positive or negative senses of the collected data. Sentiment Analysis also known as Opinion Mining refers to the use of natural language processing, text analysis and computational linguistic to identify and extract subjective information in source materials.

As most the sentiment that is uploaded to the internet is of an unstructured nature it is a difficult task for computers to process it and extract meaningful information from it. Natural language processing techniques are used to transform this raw data into a form that it can be processed efficiently by a computer.

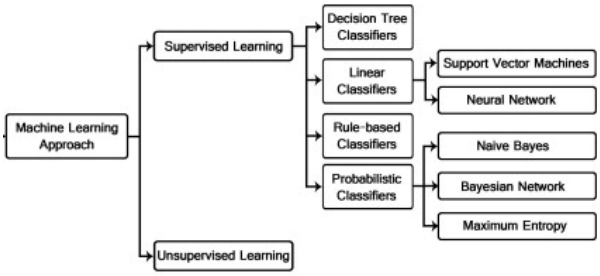
Machine Learning Algorithms

Machine learning is a branch of artificial intelligence which focuses on building models that can learn from data.

In general, there are two types of machine learning algorithms supervised, and unsupervised, there are variations on these algorithms such hybrid types semi-supervised learning but for this report they will be classified into one of the two categories.

The method of supervised learning consists of presenting an algorithm with a training dataset; this dataset consists of training examples and the corresponding expected output for each example. The expected output is general known as the target. A supervised learning algorithm uses this dataset so that it can learn to map the input examples to their expected target. If the training process is implemented correctly the machine learning algorithm should be able to generalize the training data so that it can correctly map new data that it has never seen before.

Unsupervised machine learning algorithms do not require training data; they operate on data where the output is unknown. The object of this form of learning is usually to discover patterns in the data that may not be known by the researcher. An example of an unsupervised method would be clustering where the algorithm uses a distance function to group similar data points together.



In this paper, we are using a Supervised Machine Learning algorithm i.e. Naïve Bayes to calculate the accuracies, precisions (of positive and negative corpuses) and recall values (of positive and negative corpuses).

Bayes Theorem

To understand how naive Bayes classifiers work, we must briefly recapitulate the concept of Bayes' rule. The probability model that was formulated by Thomas Bayes (1701-1761) is quite simple yet powerful; it can be written down in simple words as follows:

$$\text{posterior probability} = \frac{\text{conditional probability} \cdot \text{prior probability}}{\text{evidence}}$$

Bayes' theorem forms the core of the whole concept of naive Bayes classification. The *posterior probability*, in the context of a classification problem, can be interpreted as: "What is the probability that a particular object belongs to class i given its observed feature values?" A more concrete example would be: "What is the probability that a person has diabetes given a certain value for a pre-breakfast blood glucose measurement and a certain value for a post-breakfast blood glucose measurement?"

$$P(\text{diabetes}|xi),\ xi=[90mg/dl,145mg/dl]$$

Let

- xi be the feature vector of sample i, $i \in \{1, 2,...,n\}$,
- ωj be the notation of class j, $j \in \{1,2,...,m\}$,
- and $P(xi|\omega j)$ be the probability of observing sample xi given that is belongs to class ωj

The general notation of the posterior probability can be written as:

$$P(\omega_j \mid \mathbf{x}_i) = \frac{P(\mathbf{x}_i \mid \omega_j) \cdot P(\omega_j)}{P(\mathbf{x}_i)}$$

The objective function in the naive Bayes probability is to maximize the posterior probability given the training data to formulate the decision rule.

To continue with our example above, we can formulate the decision rule based on the posterior probabilities as follows: person has diabetes:

if P(diabetes|xi) ≥ P(not-diabetes|xi),
else classify person as healthy.

4. Related Work

Sentiment analysis of natural language texts is a large and growing field with research ranging from document level classification to learning the polarity of words and phrases. Just in the past years there have been several papers looking at Twitter sentiment.

Here are few related works that has been done in sentiment analysis:

- Alec co[4] used different machine learning algorithms such as Naïve Bayes', Support vector machine and maximum entropy.
- Janice M.[5] Weibe performed document and sentence level classification. He classified the terms into positive and negative class and calculated the positive or negative score for the text. If the overall positive score is more than negative, then the document is considered positive otherwise negative.
- Theresa Wilson, Janyce Wiebe and Paul Hoffman[7] worked on a new approach on sentiment analysis by first determining the neutrality of an expression and Tracking customer's opinions and ratings on products and services. Second, monitoring issues confronting the company so as polar and then disambiguates the polarity of the polar expression. With this approach the system can automatically identify the contextual polarity for a large subset of sentiment expressions, hence achieving results which are better than baseline.

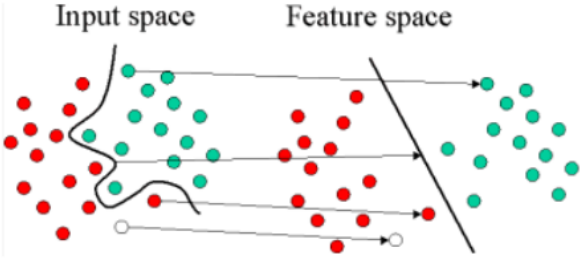
Other Methods

Here we discuss some other supervised learning algorithms that can be used for sentiment analysis.

Support Vector Machine

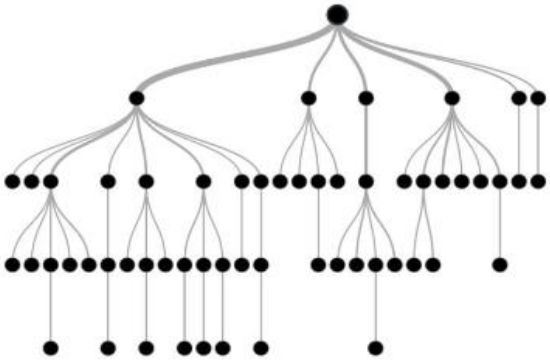
Support vector machine (SVM) is a method for the classification of both linear and nonlinear data. If the data is linearly separable, the SVM searches for the linear optimal separating hyperplane (the linear kernel), which is a decision boundary that separates data of one class from another. If the data is linearly inseparable, the SVM uses nonlinear mapping to transform the data into a higher dimension. It then solves the problem by finding a linear hyperplane.

A major problem with the SVM is that by adding extra dimensions the size of the feature space increases exponentially.



Decision tree

Decision tree classifier provides a hierarchical decomposition of the training data space in which a condition on the attribute value is used to divide the data. [9] The knowledge is then presented in the form of logical structure similar to a flow chart that can be easily understood without any statistical knowledge. This algorithm is particularly well suited to cases where many hierarchical categorical distinctions can be made.[8]



Decision Tree Structure

5. Project Description

This section aims to talk about the implementation of our project in formal detail.

Dataset

To conduct the research dataset is collected from www. http://help.sentiment140.com/for-students/ which has more than 100K comments in the terms of positive and negative comments.

Training Data

To train a supervised learning algorithm a training dataset must be collected; this dataset consists of training examples and the corresponding expected output for each example. The expected output is general known as the target. A supervised learning algorithm uses this dataset so that it can learn to map the input

examples to their expected target. If the training process is implemented correctly the machine learning algorithm should be able to generalize the training data so that it can correctly map new data that it has never seen before.

Training data must contain a class label; this can be achieved through manually assigning each tweet with a class but this is a tedious process and as twitter enforces strict rules on the distribution of its data it has proved difficult to source reliable hand annotated twitter datasets. For this reason, other avenues were examined and it was found that several researchers have successfully used emoticons ☺ ☹ as a noisy label to automatically classify sentences. To use this method an assumption must be made; this assumption is that the emoticon in the tweet represents the overall sentiment contained in that tweet. This assumption is quite reasonable as the maximum length of a tweet is 140 characters so in most cases the emoticon will correctly represent the overall sentiment of that tweet.

Emoticon Usage Frequency

Emoticon	Usage	Percentage
:)	32,115,778	33.36%
:D	10,595,385	11.006%
:(	7,613,078	7.90%
;)	7,238,729	7.519%
:-)	4,250,408	4.420%

For this project the smiley face ☺ and the sad face ☹ were chosen as the noisy labels for the training data.

Sample Training Data

Tweet	Class
Watching a nice movie ☺	Positive (1)
The painting is ugly, will return it tomorrow…	Negative (0)
One of the best soccer games, worth seeing it.	Positive (1)
Too early to travel… Need a coffee ☹	Negative (0)

Preprocessing and Data Cleaning

Due to the nature of language used in micro-blogging posts, preprocessing on the messages is very necessary and has been shown to improve performance, especially for smaller training sets. We used techniques that have already been used for this task as well as several new methods.

Preprocessing is necessary because it can extract relevant features while sanitizing inputs. Micro-blogging posts, specifically, can be full of spelling and grammar errors as well as slang and emoticons, so providing a more consistent lexical input could improve performance.

The techniques used have been applied in information retrieval applications as well as in other sentiment analysis tasks, including micro-blogging specifically. Preprocessing methods used include identifying emoticons, identifying words in all caps, detecting pointers to other users, detecting URLs, removing stop words, removing unnecessary punctuation, identifying exclamation marks or exclamation marks intertwined with question marks, identifying laughter, identifying the query term, and removing erratic casing of letters.

Emoticons – Many micro-blogging posts make use of emoticons to convey emotion, making them very useful for sentiment analysis. A range of about 30 emoticons, including :) :(:D =] :] =) =[=(are replaced with either a SMILE or FROWN keyword. In addition, variations of laughter such as “haha” or “ahahaha” are all replaced with a single LAUGH keyword.

All Caps – The use of all capital letters is another common method for indicating strong emotion. By identifying a series of capitalized words and adding an ALL_CAPS keyword, this preprocessing step extracts this feature before removing casing. Erratic Casing – To address the problem of posts containing various casings for words (e.g. “HeLLo”), we sanitized the input by lower casing all words, which provides some consistency in the lexicon.

Punctuation – In micro-blogging posts, it is common to use excessive punctuation to avoid proper grammar and to convey emotion. By identifying a series of exclamation marks or a combination of exclamation and question marks before removing all punctuation, relevant features are retained while more consistency is maintained.

Stop Words – In information retrieval, it is a common tactic to ignore very common words such as “a”, “an”,” the”, etc. since their appearance in a post does not provide any useful information in classifying a document.

Query Term – Since the query term itself should not be used to determine the sentiment of the post with respect to it, every query term is replaced with a QUERY keyword. Although this makes it somewhat of a stop word, it can still be useful when not using a bag-of-words model and the location of the query in relation to other words becomes important.

Pointers & URLs – Many micro-blog posts contain URLs to share more content than can be given in the limited post. The uniqueness of URLs makes it more useful to simply identify the use of one.

Similarly, posts can point to other users with the use of an ‘@’ in front of a username, which is replaced with a POINTER keyword.

Naïve Bayes Classification

In our proposed system, we use the Naive Bayesian classifier i.e. based on Bayes theorem with independence assumptions

between predictors. A Naive Bayesian model is easy to build, with no complicated iterative parameter estimation which makes it particularly useful for very large datasets. Despite its simplicity, the Naive Bayesian classifier often does surprisingly well and is widely used because it often outperforms more sophisticated classification methods. Naive Bayes is Bayesian probability distribution model based algorithm. In general, all Bayesian models are derivatives of the well-known Bayes Rule, which suggests that the probability of a hypothesis given a certain evidence, i.e. the posterior probability of a hypothesis, can be obtained in terms of the prior probability of the evidence, the prior probability of the hypothesis and the conditional probability of the evidence given the hypothesis. Mathematically,

$$P(H/E) = P(H)P(E/H)/(P(E)) \quad (1)$$

Where,

P(H|E) - posterior probability of the hypothesis.

P(H) - prior probability of hypothesis.

P(E) - prior probability of evidence.

P(E|H)-conditional probability of evidence of given hypothesis.

Or in a simpler form: -

Posterior= ((Prior) × (Likelihood))/Evidence (2)

To explain the concept, let's take an example. For instance, we have a new tweet to be classified in to one of the positive or negative classes. Given that in the previously classified tweets, positive tweets are twice the number of negative tweets. Since the new tweets class is not known, the problem is estimating correctly the class that the tweet is to be categorized in. This can be found out by Bayes rule calculating the probabilities of the likelihood of the tweet to be positive or negative. Hence, Design and Analysis of Proposed Approach we have:

$$P(n/p) = \frac{p(n)P\left(\frac{p}{n}\right)}{P(p)}$$

Since there are twice as many positive tweets as negative, it is reasonable to believe that a new case (which hasn't been observed yet) is twice as likely to have membership positive rather than negative. In the Bayesian analysis, this belief is known as the prior probability. Prior probabilities are based on previous experience. In this case the percentage of positive tweets and negative tweets, and often used to predict outcomes before they happen. Thus, we can write:

Prior Probability of positive tweet:

$$P(p) = \frac{\text{No.of positive tweets}}{\text{Total no.of tweets}}$$

Prior Probability of negative tweet:

$$P(n) = \frac{\text{No.of negative tweets}}{\text{Total no.of tweets}}$$

Let there be say a total of 6k tweets, 4k of which are positive and 2k negative, our prior probabilities for class membership.

Prior Probability for positive tweet P(p) = 6k/ 9k = 6/9 = 2/3

Prior Probability for negative tweet P(n)= 2k /6k = 2/ 6 =1/3

The likelihood of the tweet falling into either of the classes is equal, since we have only two classes. So, likelihood of X = 0.5. So now calculating the posterior probability of the new tweet say X, being positive or negative will be:

• Posterior probability of X being positive = (Prior probability of positive) × (Likelihood of X being positive) = 6/9 × 1/2 = 1/3 = 33.34% chances of X being positive.

• Posterior probability of X being negative = (Prior probability of negative) × (Likelihood of X being negative) = 1/3 × 1/2 = 1/6 = 16.67% chances of X being negative.

Thus, this tweet will fall in to the positive class: -

In our case, we would have two hypotheses and many other features on basis of which the one that has the highest probability would be chosen as a class of the tweet those sentiment is being predicted. Sentiment Analysis is automatic extraction of subjective content of text and predicting the subjectivity such as positive or negative.

Sentiment Analysis is the process of taking a block of text and determining if the author feels positive or negative about a topic. It can be an extremely difficult problem to do correctly. For instance, consider the following (naive) approach. This approach simply takes a dictionary of words and assigns a positive or negative weight to each. To determine the overall sentiment of a phrase, simply add up the scores of the words found in the dictionary set are shown below:

Table 1: The example for keyword and sentiment.

Keyword	Sentiment
Excellent	10
Impressed	6
Great	5
Ok	1
Meh	0
Boring	-3
Sick	-4
Terrible	-5

Now, taking this table, we can assign values to the following sentences:

- The movie was great! Excellent explosions! 5 + 10 = 15
  Here since the sentiment value of the tweet is +15, it is a positive sentiment.
- I thought the movie was terrible. Boring! -5 + (-3) = -8
  Here since the sentiment value of the tweet is -8, it is a negative sentiment.

The below Fig. 3 shows that we gather information from twitter and process data by using Naïve Bayesian classifier to divide the bunch of words into positive and negative tweets. These words form into a type of dictionary set, and that data is processed by map reduce algorithm to give more accuracy and better performance.

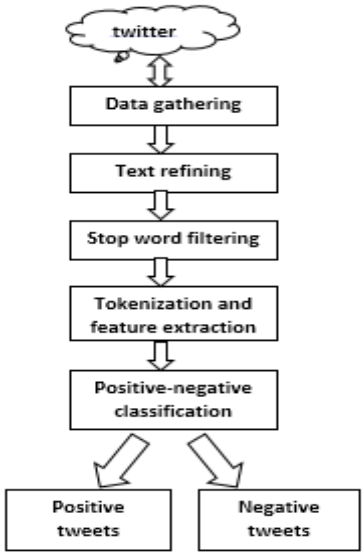


Fig 3: - Process for classification of tweets

Implementation of Classifier

We started by transforming the input file to sequence file format. This file format is used by Hadoop which is further used by Mahout for parallel processing.

```
public void inputDataToSequenceFile() throws Exception {
    BufferedReader reader = new BufferedReader(
            new FileReader(inputFilePath));
    FileSystem fs = FileSystem.getLocal(configuration);
    Path seqFilePath = new Path(sequenceFilePath);
    fs.delete(seqFilePath, false);
    SequenceFile.Writer writer = SequenceFile.createWriter(fs,
            configuration, seqFilePath, Text.class, Text.class);
    int count = 0;
    try {
        String line;
        while ((line = reader.readLine()) != null) {
            String[] tokens = line.split("\t");
            writer.append(new Text("/" + tokens[0] + "/tweet" + count++),
                    new Text(tokens[1]));
        }
    } finally {
        reader.close();
        writer.close();
    }
}
```

The next method, sequenceFileToSparseVector(), uses the previously created sequence file to create SparseVectors. These vectors contain the TFIDF measurement for the words in the tweets and will be used to train the classifier.

```
void sequenceFileToSparseVector() throws Exception {
    SparseVectorsFromSequenceFiles svfsf = new SparseVectorsFromSequenceFiles();
    svfsf.run(new String[] { "-i", sequenceFilePath, "-o", vectorsPath,
            "-ow" });
}
```

All the tweets and labels are passed to the classifier. We trained the Classifier by passing a bulk of training data tweets to the Naïve Bayesian classifier.

Then the input containing the tweets to be classified are split into several chunks. Each chunk is sent to the Hadoop node that will process it by running the Naive Bayes model and some document/word frequency into memory for each tweet of the chunk, it computes the best matching category.

The trainNaiveBayesModel() method creates the model file, staring from the TFIDF vectors. The last method, classifyNewTweet, takes a new tweet, creates the TFIDF vector from the words and calculates the probability for this tweet of being positive or negative. The highest probability decides the polarity of the tweet.

```
void trainNaiveBayesModel() throws Exception {
    TrainNaiveBayesJob trainNaiveBayes = new TrainNaiveBayesJob();
    trainNaiveBayes.setConf(configuration);
    trainNaiveBayes.run(new String[] { "-i",
            vectorsPath + "/tfidf-vectors", "-o", modelPath, "-li",
            labelIndexPath, "-el", "-c", "-ow" });
}
```

6. Experiments

The dataset we used is in text format. In the data file, each line is a single input including two parts: the label and the body of the tweet. There are two categories in our dataset, positive and negative. We used number 0 to represent negative and number 1 to represent positive.

The classifier was first trained with different percentage of training data and for each experiment accuracy of the system was calculated which is shown in table below.

Accuracy table for NBC using MapReduce with the percentage of training data

Data Set		Naïve Bayes				
% of test data	No. of tweets	TP	TN	FP	FN	accuracy
20	209715	107615	46400	37344	18356	73.44
30	314572	15767	217331	77599	3875	74.10
40	419430	210576	103494	67886	37474	74.88
50	524287	281178	115498	88560	39051	75.66

A total of 100 tweets were kept aside for accuracy testing distributed equally with respect to the two labels (0 for negative and 1 for positive). It is necessary to avoid negligible mistakes in the validation dataset as the errors in the generated parameters will be faulty and will lead to false analysis and evaluations.

Performance metrics are used for the analysis of classifier accuracy.

Accuracy: The efficiency of the used framework (Or) model in finding out the positive and negative instances is called accuracy.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Following contingency table is used to calculate the various measures:

Table 1

Detected Opinions	True Positive	False Positive
Undetected Opinions	False Negative	True Negative

Using this table, we determine the formula for calculating accuracy of the system.

7. RESULTS

Here we calculated our results based on our experiments and made several graphs showing the performance of our system.

When the dataset is relatively small, the accuracy is unstable because the training data are not big enough for the model to learn enough knowledge about each class. As the dataset size increases, the accuracy gradually climbs above 80% and approaching 82%. This demonstrates that the accuracy of Naïve Bayes' Classifier is better when training data set is large.

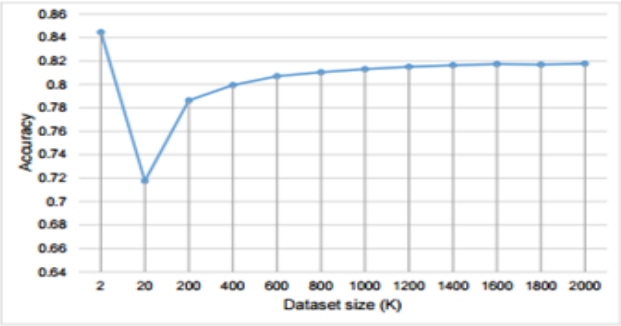


Fig: Accuracy of Naïve Bayes when corpus size increases

To further examine the classification results, we plot in Fig. 4 the breakdown of accuracy into true positive and true negative, accompanied by false positive and false negative. As we expected, the true positive and true negative increase with respect to the dataset size, while the false positive and false negative decrease.

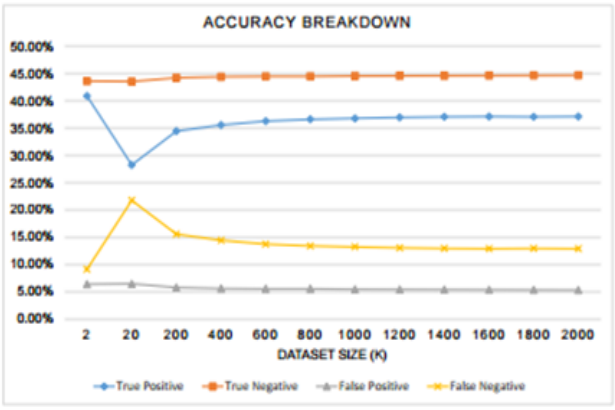


Fig: The accuracy breakdown of Naïve Bayes classifier when the Dataset size increases.

Table 2

Dataset size(K)	2	20	30	50	70	100
Seconds/10K Tweets	455.1	48.15	40.61	33.4	24.37	12.3

Table 2 shows that the processing time for every ten thousand reviews in our Hadoop NBC decreases when the dataset size increases. A dataset of 2K reviews did not benefit from the parallelization of Hadoop because the input data is smaller than the size of one block in HDFS. After the input data increasing to a certain size, the advantage of Hadoop starts to appear in that the processing time for the same amount of reviews is drastically reduced compare to the 2K case. The accuracy tends to be stable when the dataset size increases. These results are based on the simple processing of review texts. Further filtering of the input data might be able to increase the accuracy.

The performance of the system depends on training datasets and content (i.e. Tweets) in these data sets. Thus, this is very simple and effective approach to analyze the sentiment form text. If we add other classifiers such as Maximum Entropy and Ensemble classifier we can easily analyze these results.

```
Tweet: Today was a bad day ;(
Probability of being negative: -12.091594457202355
Probability of being positive: -23.08239149386524
The tweet is negative :(

Tweet: I don't like you
Probability of being negative: -31.07009731036127
Probability of being positive: -41.791680283872836
The tweet is negative :(

Tweet: I am very happy today :)
Probability of being negative: -61.17651001155036
Probability of being positive: -54.59703182199123
The tweet is positive :)
```

```
Tweet: I love the way you lie
Probability of being negative: -36.85412590563361
Probability of being positive: -26.786367000421777
The tweet is positive :)
```

Fig: Output of our Classifier on the given tweets

Manual Validation of Tweets

Tweet	Actual Class	Classified as	Correct/Incorrect
can't help but be encouraged by what he's saying DAMNIT JARED	Positive	Negative	Incorrect
Can;t wait for Football Final Tomorrow	Positive	Negative	Incorrect
Supernatural is ruining my life I hate this show send help	Negative	Negative	Correct
omg Katy Perry is SO CUTE ! I wanna thank you for this beautiful music , it was definitely worth the wait! I love you so much	Positive	Positive	Correct
I love it when Sam is a geek. reminds me of the good ol season 1 days #Supernatural	Positive	Positive	Correct
I love it when my husband and I match	Positive	Positive	Correct
It's our responsibility to be there for one another. Love each other and make our world BEAUTIFUL	Positive	Positive	Correct
OH, I GOT A LOVE THAT KEEPS ME WAITING Sherlock Season Finale TODAY	Positive	Positive	Correct
Just watched my first ep of #Flash and it was good!	Positive	Positive	Correct

Limitations of Naïve Bayes

Despite naïve Bayes technique is very simple and easy to implement. It still holds some Issues or limitations. These are:

1. Incomplete Training data*:* To implement it, we need to compute several conditional probabilities. Precisely, the class conditional probability, which defines the probability that an attribute supposes a specific value, given the consequence or reply class. In the standard naive Bayes instance of cricket data, there are no instances of "Play = No" when the trait "outlook" is "cloudy". So, the class conditional probability would be zero and the entire construction breakdowns.

2. Continuous Variables: When a characteristic is continuous, calculating the probabilities by the traditional technique of frequency counts is impossible. In this case, we would either need to transform the characteristic to a discrete variable or use probability density functions to calculate probability densities (not actual probabilities!).

3. Attribute Independence: This is by far the most important flaw and something which obliges a little bit of extra effort. In the calculation of outcome probabilities using the classical Bayes theorem, the implicit assumption is that all the traits are mutually liberated. This allows us to multiply the class conditional probabilities to compute the outcome probability.

8. Conclusion

In this project, we investigated applying sentiment analysis using Naïve Bayes to analyze sentiment for a given tweet using a collection of training data that we used to train our classifier. Using various preprocessing techniques and Naïve Bayes classifiers, we could achieve reasonably good performance for the large training set used. The experimental results show that classifier yielded good results for the review data with the Naïve Bayes' approach giving above 80% accuracies. From numerical based approach group, Naive Bayes has several advantages such as simple, fast and high accuracy. We also saw that the performance of the proposed algorithm in Hadoop is accurate and processing time is less compared with different

traditional data mining tools and techniques. The different approaches for improving pre-processing and accounting for problems with the data set as well as boosting classifier performance leaves a wide range of further improvements.

9. Future Work

Sentiment Analysis can be very effective in predicting Election results, stock market or movie review. For further work, we would like to compare try and come up with an efficient sentiment analyser like decision trees, Support vector Machine etc. and try to implement a new algorithm utilizing the benefits of the both algorithms so that it can be used effectively in data forecasting.

10. References

[1] Lina L. Dhande and Dr. Prof. Girish K. Patnaik, "Analyzing Sentiment of Movie Review Data using Naive Bayes Neural Classifier", *IJETTCS, Volume 3, Issue 4 July-August 2014, ISSN 2278-6856*.

[2] P.Kalaivani, "Sentiment Classification of Movie Reviews by supervised machine learning approaches" et.al,*Indian Journal of Computer Science and Engineering (IJCSE) ISSN* : 0976-5166 Vol. 4 No.4 Aug-Sep 2013.

[3] Meena Rambocas, João Gama, "Marketing Research: The Role of Sentiment Analysis", *April 2013, ISSN: 0870-8541*.

[4] Weiguo Fan, Linda Wallace, Stephanie Rich, and Zhongju Zhang, "Tapping into the Power of Text Mining", *Journal of ACM, Blacksburg, 2005*.

[5] "Movie review dataset," [Online]. Available *http://www.cs.cornell.edu/people/pabo/movie-reviewdata/*, [Accessed: October 2013].

[6] K. M. Leung, "Naive Bayesian classifier," [Online] Available: *http://www.sharepdf.com/81fb247fa7c54680a94dc0f3a253fd85/naiveBayesianClassifier.pdf, [Accessed: September 2013]*.

[7] Zhou Yong , Li Youwen and Xia Shixiong "An Improved KNN Text Classification Algorithm Based

[8] John dudd, "Twitter sentiment Analysis" 2014

[9] J.R. Quinlan Induction of decision trees Machine Learn, 1 (1986), pp. 81–106

[10] I. Rish, "An empirical study of the naive bayes classifier," in IJCAI 2001 workshop on empirical methods in artificial intelligence, pp. 41–46, 2001.

[11] P. Domingos and M. Pazzani, "On the optimality of the simple bayesian classifier under zero-one loss," Machine learning, vol. 29, no. 2-3, pp. 103–130, 1997.

[12] J. Kazmierska and J. Malicki, "Application of the na¨ive bayesian classifier to optimize treatment deci- sions," Radiotherapy and Oncology, vol. 86, no. 2, pp. 211–216, 2008.

[13] Q. Wang, G. M. Garrity, J. M. Tiedje, and J. R. Cole, "Naive bayesian classifier for rapid assignment of rrna sequences into the new bacterial taxonomy," Applied and environmental microbiology, vol. 73, no. 16, pp. 5261–5267, 2007. e}

[14] M. Sahami, S. Dumais, D. Heckerman, and E. Horvitz, "A bayesian approach to filtering junk e-mail," in Learning for Text Categorization: Papers from the 1998 workshop, vol. 62, pp. 98–105, 1998.

[15] H. Zhang, "The optimality of naive bayes," AA, vol. 1, no. 2, p. 3, 2004.

[16] C. Yao, X. Zhang, X. Bai, W. Liu, Y. Ma, and Z. Tu, "Rotation-invariant features for multi-oriented text detection in natural images," PloS one, vol. 8, no. 8, p. e70173, 2013.

[17] M. F. Porter, "An algorithm for suffix stripping," Program: electronic library and information systems, vol. 14, no. 3, pp. 130–137, 1980. orithm}

[18] M. Toman, R. Tesar, and K. Jezek, "Influence of word normalization on text classification," Proceedings of InSciT, pp. 354–358, 2006.

[19] A. Zevcevic, "N-gram based text classification according to authorship," in Student Research Workshop, pp. 145–149, 2011.

[20] V. Keˇselj, F. Peng, N. Cercone, and C. Thomas, "N-gram-based author profiles for authorship attribution," in Proceedings of the conference pacific association for computational linguistics, PACLING, vol. 3, pp. 255–264, 2003.

[21] I. Kanaris, K. Kanaris, I. Houvardas, and E. Stamatatos, "Words versus character n-grams for anti-spam filtering," International Journal on Artificial Intelligence Tools, vol. 16, no. 06, pp. 1047–1067, 2007.

[21] A. McCallum, K. Nigam, et al., "A comparison of event models for naive bayes text classification," in AAAI-98 workshop on learning for text categorization, vol. 752, pp. 41–48, Citeseer, 1998.

[22] L. M. Rudner and T. Liang, "Automated essay scoring using bayes' theorem," The Journal of Technology, Learning and Assessment, vol. 1, no. 2, 2002.

[23] Apoorv Agarwal, Boyi Xie, Ilia Vovsha, Owen Rambow, Rebecca Passonneau, "Sentiment Analysis of Twitter Data" Department of Computer Science, Columbia University, New York, 2009.